VISUAL BASIC 6.0

ПОВНИЙ САМОВЧИТЕЛЬ

Від першої форми до Windows API, баз даних, COM та ActiveX

Практичний навчальний посібник

Українською мовою • VB6

Передмова

Ця книга побудована як самовчитель: від абсолютного нуля до створення повноцінних Windows-застосунків на Visual Basic 6.0. Тобі не потрібно знати програмування наперед. Достатньо розуміти базові дії у Windows, уміти створювати папки та працювати з файлами.

VB6 давно вважається історичною технологією, але його модель особливо зручна для навчання: форма, елемент керування, властивість, метод і подія дають дуже наочний шлях від ідеї до працюючої програми.

У книзі навмисно багато коду. Програмування вчиться не тільки читанням: після кожного великого розділу є практика, а в кінці зібрані проєкти, які можна реалізувати самостійно.

Примітка: VB6 працює у старому Windows-стеку. Для навчання важливо відокремлювати мову, її синтаксис та об’єктну модель від конкретних сучасних вимог до безпеки, підпису коду та сумісності.

Як користуватися книгою

1. Читай розділ послідовно, але обов’язково повторюй приклади руками.
2. Після кожного прикладу зміни одну деталь: назву, число, умову, текст або властивість.
3. Не копіюй великий проєкт цілком. Спочатку зрозумій його частини, потім збирай разом.
4. Помилки компілятора та runtime-помилки використовуй як підказки: дивись номер, опис та рядок, на якому зупинилась програма.
5. До WinAPI та COM переходь після того, як упевнено володієш змінними, процедурами, файлами, класами та базами даних.

Зміст

Частина I. Основи Visual Basic 6.0

1. Що таке VB6 і як мислити подіями

2. Середовище розробки

3. Перша програма

4. Об’єкти, властивості, методи, події

5. Змінні та типи даних

6. Оператори та вирази

7. Умови

8. Цикли

9. Масиви

10. Рядки

11. Функції та процедури

12. Область видимості, модулі та константи

13. Перетворення типів, Null та Variant

14. Дата і час

15. Обробка помилок

Частина II. Графічний інтерфейс

16. Form

17. Label, TextBox, CommandButton

18. CheckBox, OptionButton, Frame

19. ListBox і ComboBox

20. PictureBox та Image

21. Timer

22. Меню, діалоги та стандартні контроли

Частина III. Файли і дані

23. Текстові файли

24. Binary та Random Access

25. Файлова система

26. INI та конфігурація

27. Реєстр Windows

Частина IV. Об’єктна модель

28. User Defined Types

29. Enum

30. Collection

31. Class Modules

32. Property та інкапсуляція

33. Компонування класів та подієва модель

Частина V. Бази даних

34. DAO та ADO

35. Connection та Recordset

36. SQL

37. CRUD

38. Пошук, сортування, фільтрація

39. Транзакції, помилки та звільнення ресурсів

Частина VI. Windows API

40. Що таке WinAPI

41. Declare, типи та константи

42. Вікна та повідомлення

43. Shell, Clipboard та системні можливості

44. Реєстр через API

45. Таймери, клавіатура та миша

46. Системний трей та утиліти

Частина VII. Графіка та ігри

47. Система координат

48. Line, Circle, PSet та PaintPicture

49. Анімація та game loop

50. Зіткнення, стан гри та простий Pong

Частина VIII. Автоматизація та компоненти

51. Automation

52. Word та Excel Automation

53. COM

54. ActiveX DLL

55. ActiveX EXE та OCX

56. DLL і взаємодія компонентів

Частина IX. Мережа та сучасні формати

57. Winsock

58. HTTP на концептуальному рівні

59. XML та JSON

60. Кодування, Unicode та сумісність

Частина X. Налагодження і великі проєкти

61. Debugger

62. Оптимізація

63. Архітектура

64. Безпечна робота з вводом

65. Пакування та розповсюдження

66. Підсумковий проєкт: менеджер клієнтів

Частина I. Основи Visual Basic 6.0

1. Що таке VB6 і як мислити подіями

Visual Basic 6.0 — середовище розробки та мова програмування, орієнтована на швидке створення Windows-програм. Його головна навчальна цінність — подієва модель: програма реагує на дії користувача та системи.

Подієва модель

Користувач натискає кнопку → Windows повідомляє застосунок → VB6 запускає обробник Click → твій код змінює стан інтерфейсу.

Тому центральне питання у VB6 часто не «де наступний рядок», а «яка подія повинна запускати цю дію?».

Три рівні мислення

Інтерфейс — що бачить користувач. Логіка — що повинна зробити програма. Дані — що потрібно зберегти або обробити. Навіть у маленьких програмах корисно розділяти ці рівні.

Перші терміни

Object — об'єкт. Property — властивість. Method — метод. Event — подія. Control — елемент керування на формі. Form — вікно/контейнер інтерфейсу.

Практика

6. Назви по три приклади об'єктів, властивостей, методів та подій.
7. Опиши словами подію натискання кнопки без коду.
8. Визнач, які дані потрібні програмі «Список покупок».

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

2. Середовище розробки

IDE VB6 складається з форми, Toolbox, Properties, Project Explorer та редактора коду. Найважливіша звичка — постійно бачити зв'язок між елементом на формі та його іменем у коді.

Project Explorer

Типовий проєкт містить форми (.frm), модулі (.bas), класи (.cls) та файл проєкту (.vbp).

Properties

Name — ім'я, яке використовує код. Caption — видимий текст у багатьох контролах. Visible — видимість. Enabled — доступність. Left, Top, Width, Height — геометрія.

Перші гарячі звички

Зберігай проєкт регулярно. Давай об'єктам осмислені імена. Не залишай десятки control1, text2, command7 без причини.

Практика

9. Створи порожню форму та зміни її Caption.
10. Додай Label і TextBox, зміни їхні Name та Caption.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

3. Перша програма

Першу програму краще робити маленькою: одне вікно, один ввід, одна кнопка і один результат.

Приклад

Створи Text1, Label1 та Command1. У Click напиши код нижче.

Розбір

Text1.Text читає текст. Label1.Caption записує текст у Label. `Private Sub ... End Sub` — процедура-обробник події.

Практичний приклад коду

```
Option Explicit

Private Sub Command1_Click()
    Dim Name As String

    Name = Trim$(Text1.Text)

    If Len(Name) = 0 Then
        MsgBox "Введіть ім'я."
        Text1.SetFocus
        Exit Sub
    End If

    Label1.Caption = "Привіт, " & Name & "!"
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

11. Змініть програму так, щоб вона показувала «Привіт, <ім’я>!».
12. Додайте перевірку порожнього вводу.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

4. Об’єкти, властивості, методи, події

Ця четвірка — фундамент VB6. Один і той самий об’єкт має стан (properties), може виконувати дії (methods) і реагує на events.

Властивість

Присвоєння: Text1.Text = "Hello". Читання: S = Text1.Text.

Метод

Text1.SetFocus передає фокус. List1.Clear очищає список. Picture1.Refresh оновлює відображення.

Подія

Form_Load — завантаження форми. Command1_Click — натискання кнопки. Text1_Change — зміна тексту.

Практика

13. Для Button, TextBox і Form запиши по 3 властивості, 2 методи та 2 події.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

5. Змінні та типи даних

Змінна — іменована область даних. Хороша програма явно описує, які значення в ній зберігаються.

Основні типи

Boolean — True/False.

Byte — 0..255.

Integer — 16-бітове ціле.

Long — 32-бітове ціле.

Single і Double — числа з плаваючою крапкою.

Currency — точний фінансовий тип з фіксованою десятковою частиною.

Date — дата/час.

String — текст.

Object — посилання на об’єкт.

Variant — контейнер, здатний зберігати різні типи.

Оголошення

Рекомендуй собі правило: одна змінна — один зрозумілий тип. `Dim Age As Integer` краще за безіменний Variant, коли тип наперед відомий.

Option Explicit

Додавай `Option Explicit` на початку модулів, щоб компілятор вимагав оголошення змінних. Це ловить помилки друку в іменах.

Практичний приклад коду

```
Option Explicit


Private UserName As String
Private Age As Integer
Private Salary As Currency
Private IsActive As Boolean
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

14. Зроби програму, яка зберігає ім’я, вік, зарплату та прапорець активності користувача.

15. Порівняй Integer і Long на прикладах великих чисел.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

6. Оператори та вирази

VB6 має арифметичні, порівняльні та логічні оператори. Важливо відрізняти `/` від `\` та розуміти пріоритет операцій.

Арифметика

`+`, `-`, `*`, `/`, `\`, `Mod`, `^`.

Порівняння

`=`, `<>`, `<`, `>`, `<=`, `>=`.

Логіка

`And`, `Or`, `Not`, `Xor`. Використовуй дужки, коли без них вираз стає нечітким.

Рядки

Для склеювання тексту використовується `&`: `FullName = FirstName & " " & LastName`.

Практичний приклад коду

```
Dim A As Double
Dim B As Double
Dim Result As Double

A = 10
B = 3

Result = A / B
Debug.Print Result
Debug.Print 10 \ 3
Debug.Print 10 Mod 3
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

16. Напиши формули для площі кола, прямокутника та трикутника.
17. Зроби перевірку віку через And/Or.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

7. Умови

Умова дозволяє програмі вибрати одну з гілок.

If

If ... Then ... End If. Додавай Else/ElseIf тільки там, де вони роблять код зрозумілішим.

Select Case

Зручний для вибору з кількох значень або діапазонів.

Захист вводу

Перед арифметикою перевіряй IsNumeric. Перед діленням перевіряй дільник на нуль.

Практичний приклад коду

```
If Not IsNumeric(Text1.Text) Then
    MsgBox "Потрібно ввести число."
    Exit Sub
End If

If CDbl(Text1.Text) > 100 Then
    MsgBox "Велике число"
ElseIf CDbl(Text1.Text) > 0 Then
    MsgBox "Додатне число"
Else
    MsgBox "Нуль або від'ємне"
End If
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

18. Напиши програму визначення категорії віку.
19. Створи меню операцій через Select Case.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

8. Цикли

Цикли повторюють код. Обирай цикл за логікою задачі, а не за звичкою.

For

Для відомої кількості повторів. Можна Step -1 або інший крок.

Do While / Do Until

Для умовного завершення. Слідкуй, щоб умова колись стала хибною, інакше отримаєш нескінченний цикл.

For Each

Зручно для Collection та інших перелічуваних об’єктів.

Практичний приклад коду

```
Dim I As Long
Dim Total As Long

For I = 1 To 100
    Total = Total + I
Next I

Debug.Print Total
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

20. Виведи числа 1..100.
21. Виведи лише парні числа.
22. Порахуй суму 1..N.
23. Порахуй кількість цифр у числі.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

9. Масиви

Масив зберігає багато елементів одного типу під одним ім’ям.

Статичний масив

`Dim Numbers(1 To 10) As Long`. Доступ — `Numbers(I)`.

Динамічний масив

`Dim Items() As String` та `ReDim Items(1 To N)`. `ReDim Preserve` розширює масив зі збереженням даних.

Типові задачі

Пошук максимуму, мінімуму, середнього, сортування, фільтрація. Для двовимірного масиву використовуй два індекси.

Практика

24. Знайди максимум і мінімум масиву.

25. Зроби реверс масиву.
26. Напиши просте сортування бульбашкою.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

10. Рядки

Робота з рядками потрібна майже в кожній прикладній програмі.

Основні функції

Len, Left$, Right$, Mid$, InStr, Replace, Trim$, LTrim$, RTrim$, UCase$, LCase$, Space$, String$.

Побудова рядків

Використовуй `&` і функції форматування. Уникай непомітних Variant-конверсій.

Розбір

Для простих форматів використовуй Split, якщо він доступний у твоєму VB6 runtime/бібліотеках; інакше реалізуй розбір вручну через InStr та Mid$.

Практичний приклад коду

```
Dim Text As String
Dim Pos As Long

Text = Trim$(Text1.Text)
Pos = InStr(1, Text, "VB", vbTextCompare)

If Pos > 0 Then
    MsgBox "Знайдено з позиції " & Pos
End If
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім'я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

27. Зроби лічильник символів і слів.
28. Знайди всі входження підрядка.
29. Замініть у тексті всі подвійні пробіли на один.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

11. Функції та процедури

Функції повертають результат; Sub-процедури виконують дію. Винось повторювану логіку.

Function
Ім’я функції у тілі отримує значення результату.

Sub
Добре підходить для дій, які не повинні повертати значення.

Параметри
ByVal передає значення; ByRef дозволяє змінити аргумент у виклику. За замовчуванням параметри у VB6 передаються ByRef, тому явне `ByVal` часто робить намір зрозумілішим.

Практичний приклад коду
```
Private Function AddNumbers(ByVal A As Double, ByVal B As Double) As Double
    AddNumbers = A + B
End Function

Private Sub ShowTotal(ByVal Value As Double)
    Label1.Caption = Format$(Value, "0.00")
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика
30. Напиши Add, IsEven, MaxOfTwo.
31. Зроби процедуру ClearForm, яка очищає кілька полів.

Що запам’ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

12. Область видимості, модулі та константи
Розміщення даних впливає на структуру програми.

Локальні змінні
Dim усередині процедури бачиться тільки там.

Рівень модуля
Private у формі/модулі доступний у межах модуля; Public у стандартному модулі доступний ширше.

Const
`Const TAX As Double = 0.20` робить названу константу.

Модулі
Стандартний .bas добре підходить для загальних функцій; форма — для UI-логіки; клас — для поведінки об’єкта.

Практика

32. Розділи калькулятор на UI-модуль та модуль математичних функцій.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

13. Перетворення типів, Null та Variant

Variant зручний, але приховує тип. Явні конверсії роблять код зрозумілішим.

Конверсії

CInt, CLng, CSng, CDbl, CCur, CStr, CDate, CBool.

Перевірки

IsNumeric, IsDate, IsEmpty та IsNull мають різне призначення. Null — не просто порожній рядок; він означає відсутність значення у певних контекстах, особливо у БД.

Практичне правило

Перед C... із тексту перевіряй ввід. Пам’ятай про локальні десяткові роздільники та формати дат.

Практика

33. Зроби функцію SafeDoubleFromText.
34. Покажи різницю між "", Empty і Null у тестовій програмі.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

14. Дата і час

VB6 має тип Date та функції для отримання, форматування й арифметики дат.

Поточна дата

Date — дата. Time — час. Now — дата і час.

Форматування

Format$(Now, "dd.mm.yyyy hh:nn:ss") — зручний приклад; у масці хвилини позначаються `nn`, бо `mm` використовується для місяця.

Арифметика

DateAdd додає інтервал, DateDiff рахує різницю, DatePart витягає частину. Будь уважним до локалі та часового поясу, коли дані виходять за межі одного комп’ютера.

Практичний приклад коду

```
Private Sub Timer1_Timer()
    Label1.Caption = Format$(Now, "dd.mm.yyyy hh:nn:ss")
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім'я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

35. Створи цифровий годинник.
36. Порахуй кількість днів до заданої дати.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

15. Обробка помилок

Помилки бувають очікуваними та програмними. У VB6 важливо не приховувати проблему без потреби.

GoTo handler

`On Error GoTo ErrorHandler` дозволяє перейти у централізований обробник. Після успішного коду став `Exit Sub/Function`, щоб не провалитися у handler.

Resume Next

`On Error Resume Next` продовжує виконання після помилки; застосовуй вузько й перевіряй Err.Number/Description.

Err

Err.Number, Err.Description, Err.Source дають діагностичну інформацію.

Ресурси

Файли, Recordset та Connection повинні закриватися навіть коли виникла помилка.

Практичний приклад коду

```
Private Sub ReadValue()
    On Error GoTo ErrorHandler

    Dim N As Long
    N = CLng(Text1.Text)
    MsgBox N

    Exit Sub

ErrorHandler:
    MsgBox "Помилка " & Err.Number & ": " & Err.Description
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

37. Напиши безпечне читання текстового файла.
38. Створи власний ErrorHandler для калькулятора.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина II. Графічний інтерфейс

16. Form

Форма — контейнер інтерфейсу. Вона має властивості, події та методи.

Життєвий цикл

Form_Load відбувається під час завантаження. Form_Unload/QueryUnload дозволяють реагувати на закриття.

Геометрія

Left, Top, Width, Height залежать від ScaleMode та контейнера. Внутрішні координати контролів задаються відносно батьківського об’єкта.

Видимість

Visible та Enabled — два різні стани: невидимий об’єкт не показується, вимкнений показується, але недоступний.

Практика

39. Зроби форму налаштувань, яка під час Load заповнює всі елементи.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

17. Label, TextBox, CommandButton

Три базові контролли для більшості навчальних програм.

Label

Caption показує текст; Label зазвичай не редагує користувач.

TextBox

Text містить введений рядок. MultiLine, ScrollBars, MaxLength та PasswordChar змінюють поведінку.

CommandButton

Caption — текст кнопки; Click — основна подія. Default та Cancel можуть керувати Enter/Esc на формі.

Практика

40. Зроби форму авторизації без реального збереження пароля.
41. Зроби форму калькулятора з перевіркою вводу.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

18. CheckBox, OptionButton, Frame

Елементи вибору мають просту логіку.

CheckBox

Value: vbChecked, vbUnchecked, а в деяких режимах може бути vbGrayed.

OptionButton

Зазвичай об'єднуй через Frame, щоб кожна група дозволяла один вибір.

Frame

Візуальний і логічний контейнер для контролів.

Практика

42. Створи форму налаштувань з двома групами OptionButton та кількома CheckBox.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

19. ListBox і ComboBox

Списки потрібні для вибору одного чи багатьох елементів.

ListBox

AddItem, RemoveItem, Clear, ListCount, ListIndex, List(index).

ComboBox

Може мати список та поле введення залежно від Style. Text повертає поточне значення.

MultiSelect

ListBox може дозволяти множинний вибір; тоді перевіряй Selected(i).

Практичний приклад коду

```
Private Sub Command1_Click()
    Dim Text As String
```

```
    Text = Trim$(Text1.Text)
    If Len(Text) = 0 Then Exit Sub

    List1.AddItem Text
    Text1.Text = ""
End Sub

Private Sub Command2_Click()
    If List1.ListIndex >= 0 Then
        List1.RemoveItem List1.ListIndex
    End If
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

43. Зроби менеджер списку: додавання, видалення, очищення, сортування вручну.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

20. PictureBox та Image

PictureBox — універсальніший контейнер для графіки та малювання; Image легший для простих картинок.

PictureBox

Має внутрішню систему координат і графічні методи. AutoRedraw важливий для повторного відображення намальованого.

Image

Добре підходить для показу готового зображення. Stretch змінює масштаб відображення.

Практика

44. Намалюй координатну сітку у PictureBox.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

21. Timer

Timer генерує подію через заданий Interval у мілісекундах. Це зручний інструмент для простих анімацій, годинника та навчальних ігор.

Обмеження
Timer не є справжнім високоточним real-time scheduler. Не блокуй довго його обробник, інакше накопичаться затримки.

Приклад
Оновлюй Label раз на секунду, а для анімації використовуй короткий інтервал та невеликі кроки.

Практика
45. Створи секундомір.
46. Створи анімацію рухомого об’єкта.

Що запам’ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

22. Меню, діалоги та стандартні контроли
Меню і стандартні діалоги роблять програму схожою на звичний Windows-застосунок.

Меню
Menu Editor створює ієрархію пунктів, кожен пункт має Click. Винось повторювані команди в окремі процедури.

CommonDialog
Дозволяє показувати діалоги відкриття/збереження, друку, кольору та шрифту залежно від налаштувань.

Toolbar/StatusBar
Використовуються для команд і короткого стану. Їх часто поєднують з меню, щоб одна дія була доступна кількома шляхами.

Практика
47. Зроби меню File → New/Open/Save/Exit для текстового редактора.

Що запам’ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина III. Файли і дані

23. Текстові файли
Текстовий I/O у VB6 простий і підходить для конфігурації, логів та невеликих списків.

Режими
Output перезаписує файл, Append додає в кінець, Input читає.

Запис
FreeFile дає вільний номер файла; Open/Print#/Close утворюють базовий цикл роботи.

Читання
EOF перевіряє кінець, Line Input # читає рядок без розбору роздільниками. Input # працює з форматом, який VB6 очікує як дані.

Практичний приклад коду
```
Private Sub SaveText(ByVal FileName As String, ByVal Text As String)
    Dim F As Integer
    F = FreeFile

    Open FileName For Output As #F
    Print #F, Text
    Close #F
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика
48. Зроби лог-файл з датою, часом та повідомленням.
49. Збережи список справ у текстовому файлі.

Що запам’ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

24. Binary та Random Access
Двійкові та random-access режими потрібні, коли важлива структура записів або точний контроль байтів.

Binary
Get/Put працюють з блоками даних і змінними/масивами.

Random Access
Type-структури можуть зберігатися записами заданої довжини. Цей підхід історично зручний для простих локальних БД.

Застереження
Не змішуй довільні текстові записи та binary-структури без чіткого формату. Документуй версію формату файла.

Практика
50. Збережи масив структур User у random-access файл.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

25. Файлова система

VB6 має функції Dir, Kill, MkDir, RmDir, Name, FileCopy для базових операцій.

Перевірки

Dir$ повертає ім’я знайденого файла; для складніших задач зручні FileSystemObject через Scripting Runtime.

Шляхи

Не будуй шляхи простим складанням із випадкових слешів; використовуй зрозумілу функцію PathJoin або постійний роздільник `\`.

Безпека

Не довіряй шляхам, які вводить користувач. Перед Kill/Name перевіряй, що саме буде змінено.

Практика

51. Зроби простий менеджер файлів: список, розмір, перейменування, копіювання.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

26. INI та конфігурація

Для невеликих програм корисно мати текстову конфігурацію з секціями та ключами.

Формат

[General]
Language=uk
Theme=Classic — простий, читабельний формат.

Підхід

Тримай читання/запис налаштувань в одному модулі. При запуску — завантажуй у пам’ять, під час зміни — оновлюй, при виході — зберігай.

Альтернатива

Registry може бути доречним для Windows-налаштувань, але для переносимості текстовий файл часто простіший.

Практика

52. Реалізуй GetSettingValue/SetSettingValue для INI-файла без зовнішньої бібліотеки.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

27. Реєстр Windows

VB6 може працювати з Windows Registry через API або готові механізми/компоненти.

Що зберігають

Налаштування програми, асоціації, параметри інтеграції.

Гілки

HKCU — налаштування поточного користувача, HKLM — системні налаштування, доступ до HKLM може вимагати підвищених прав.

Правило

Не використовуйте Registry як довільну базу даних. Для великих даних краще файл або БД.

Практика

53. Знайди та прочитай тестовий параметр у HKCU.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина IV. Об'єктна модель

28. User Defined Types

Type дозволяє об'єднати кілька полів у структуру.

Приклад

```
Type TUser
 Name As String
 Age As Integer
End Type
```

Коли використовувати

Для простих пакетів даних. Для поведінки та методів краще Class Module.

Практика

54. Створи структури Product та Order.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

29. Enum

Enum називає числові стани зрозумілими іменами.

Приклад

```
Public Enum AppMode
 ModeNormal = 0
 ModeAdmin = 1
End Enum
```

Користь

Менше «магічних чисел», легше читати Select Case та умови.

Практика

55. Створи Enum для стану замовлення.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

30. Collection

Collection — вбудований контейнер з індексом та необов'язковим ключем.

Операції

Add, Remove, Count, Item.

Коли зручно

Для невеликих списків об'єктів, коли не потрібен складний словник.

Практика

56. Створи Collection клієнтів і пошук за ключем.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

31. Class Modules

Class Module — основа ООП у VB6. Він може мати Private поля, Public методи і властивості.

Приклад

Клас User приховує mName та відкриває Property Get/Let Name.

Інстанціювання

Dim U As User або Dim U As New User. Явне Set із New часто робить момент створення зрозумілішим.

Initialize/Terminate

Class_Initialize та Class_Terminate дають місця для створення/звільнення ресурсів. Не покладайся на Terminate для критичних операцій, якщо життєвий цикл не контрольований.

Практичний приклад коду

```
Option Explicit

Private mName As String
Private mAge As Long

Public Property Get Name() As String
    Name = mName
End Property

Public Property Let Name(ByVal Value As String)
    mName = Value
End Property

Public Property Get Age() As Long
    Age = mAge
End Property

Public Property Let Age(ByVal Value As Long)
    If Value < 0 Then Err.Raise 5, , "Вік не може бути від'ємним."
    mAge = Value
End Property
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

57. Створи clsCustomer, clsProduct, clsOrder.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

32. Property та інкапсуляція

Властивості дозволяють приховати внутрішнє поле та контролювати зміну значень.

Property Get/Let

Get читає, Let присвоює значення для простих типів. Set використовується для об’єктних посилань.

Валідація

Property Let Age може заборонити від’ємний вік.

Інкапсуляція
Код зовні не повинен знати, як саме зберігається значення.

Практика
58. Реалізуй властивість Price з перевіркою на від'ємне число.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

33. Компонування класів та подієва модель
Складні об'єкти будуються з інших об'єктів. VB6 також підтримує події класів через WithEvents/декларації подій у відповідних сценаріях.

Композиція
Order може містити Customer та Collection ProductLine.

Інтерфейс
Винось методи, які потрібні користувачу класу; внутрішні деталі залишай Private.

Події
Подія добре підходить, коли об'єкт повідомляє UI про зміну стану, а UI не повинен постійно опитувати об'єкт.

Практика
59. Спроєктуй модель для програми обліку замовлень.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина V. Бази даних

34. DAO та ADO
VB6 історично працював з базами через DAO, ADO та інші провайдери. Вибір залежить від джерела даних та старого коду, який потрібно підтримувати.

DAO
Особливо тісно пов'язаний з Access/Jet-екосистемою старих застосунків.

ADO
Більш загальна модель через Connection, Command, Recordset та провайдер.

Практика
В одному проєкті обери основний стек і не змішуй підходи без причини.

Практика

60. Поясни різницю між Connection і Recordset.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

35. Connection та Recordset

Connection описує канал до БД; Recordset представляє набір записів.

Життєвий цикл

Open → робота → Close. Після використання звільняй об’єкти.

Поля

rs.Fields("Name").Value. Перевіряй EOF/BOF.

Переходи

MoveFirst, MoveNext, MovePrevious, MoveLast; для порожнього набору будь обережним.

Практика

61. Завантаж 10 клієнтів у ListBox.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

36. SQL

SQL — мова запитів до реляційних БД.

SELECT

Вибірка: SELECT Name, Email FROM Customers WHERE Active = 1 ORDER BY Name;

INSERT/UPDATE/DELETE

INSERT додає, UPDATE змінює, DELETE видаляє. Усі три потребують уважного WHERE, коли операція не повинна торкнутися всіх рядків.

Параметри

Для реального застосунку не складай SQL через безмежну конкатенацію введення користувача. Використовуй параметризовані Command/Query там, де це підтримує провайдер.

Практичний приклад коду

```
SELECT ID, Name, Email
FROM Customers
WHERE Active = 1
ORDER BY Name;
```

```
INSERT INTO Customers (Name, Email)
VALUES (?, ?);
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім'я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

62. Напиши SQL для списку клієнтів за містом.
63. Зроби UPDATE одного клієнта.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

37. CRUD

CRUD = Create, Read, Update, Delete — базовий набір операцій для прикладної програми.

Форма

Створення — поля + Save. Читання — список/пошук. Оновлення — вибір запису + Edit. Видалення — Delete із підтвердженням.

Валідація

Перевіряй обов'язкові поля до відправки запиту.

Практика

64. Зроби повний CRUD для таблиці Customers.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

38. Пошук, сортування, фільтрація

Розділяй фільтрацію на рівень SQL і UI.

SQL-фільтр

Коли даних багато, нехай БД відфільтрує їх через WHERE.

Сортування

ORDER BY краще за завантаження всіх рядків і сортування в пам'яті, якщо даних багато.

UI

Пошук з TextBox + Button/Timer, але не запускай важкий запит на кожен символ без потреби.

Практика

65. Створи пошук клієнтів за прізвищем.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

39. Транзакції, помилки та звільнення ресурсів

Транзакція дозволяє виконати кілька змін як одну логічну операцію.

Transaction

BeginTrans → Execute ... → CommitTrans; у разі помилки — RollbackTrans.

Ресурси

Закрий Recordset перед Connection; звільни Set rs = Nothing / Set cn = Nothing.

Помилки

Логуй SQL/номер помилки без включення чутливих секретів та паролів.

Практика

66. Створи транзакцію для замовлення та його рядків.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина VI. Windows API

40. Що таке WinAPI

WinAPI — набір системних функцій Windows. VB6 може оголошувати та викликати частину цих функцій.

Чому це потрібно

Щоб робити те, що не має прямого VB6-контролу: отримувати системну інформацію, керувати вікнами, працювати з повідомленнями та ін.

Ризики

Неправильний тип параметра або структура може привести не просто до помилки, а до аварійного завершення процесу. API-код потрібно тестувати окремо.

Практичний приклад коду

```
Private Declare Function GetTickCount Lib "kernel32" () As Long


Private Sub Command1_Click()
    Debug.Print GetTickCount()
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

67. Назви три задачі, для яких API може бути доречним.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

41. Declare, типи та константи

Оголошення API потребує точного підпису функції.

Declare

Private Declare Function GetTickCount Lib "kernel32" () As Long

Типи

DWORD/BOOL/HANDLE та інші C-типи часто маплятся на Long у старій 32-бітній моделі. Але під час портів треба перевіряти розрядність та конкретний підпис.

Константи

API-код читається краще з іменованими константами.

Практика

68. Зроби оболонку VB6 навколо одного безпечного API-виклику.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

42. Вікна та повідомлення

Windows багато в чому працює через дескриптори (hWnd) та повідомлення.

hWnd

Дескриптор — число, що ідентифікує вікно для API. Не плутай hWnd із власним Name контролу.

Messages

WM_... — системні повідомлення. Subclassing дає змогу перехоплювати їх, але це вже просунутий рівень і потребує обережності.

FindWindow/SendMessage

Такі API-функції можуть бути корисними для інтеграції, але не варто робити нестабільну логіку залежною від назв чужих вікон.

Практика

69. Знайди просте вікно власної тестової програми через hWnd.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

43. Shell, Clipboard та системні можливості

VB6 має ряд вбудованих механізмів, які закривають типові сценарії.

Shell

Shell запускає процес. Для запуску файла стандартним застосунком часто зручніший ShellExecute API.

Clipboard

Clipboard.Clear, SetText, GetText.

App/Environ

App.Path дає шлях застосунку; Environ$ читає змінні середовища.

Практичний приклад коду

```
Private Sub Command1_Click()
    Clipboard.Clear
    Clipboard.SetText Text1.Text
End Sub

Private Sub Command2_Click()
    If Clipboard.GetFormat(vbCFText) Then
        Text1.Text = Clipboard.GetText()
    End If
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

70. Зроби кнопку «Відкрити папку програми».

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

44. Реєстр через API

Для низькорівневого контролю Registry використовуються RegOpenKeyEx, RegQueryValueEx та RegSetValueEx та інші функції.

Структура
Root key → subkey → value.

Типи
REG_SZ — рядок; REG_DWORD — 32-бітове число; існують інші типи.

Практика
Читай і записуй лише власні ключі програми. Не змінюй системні гілки без чіткої причини та перевірки прав.

Практика
71. Створи власний тестовий ключ у HKCU і запиши рядок.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

45. Таймери, клавіатура та миша
API дає доступ до низькорівневих подій.

Таймер
SetTimer/KillTimer працюють на рівні Windows messages. Для простих форм часто достатньо Timer control.

Клавіатура/миша
GetAsyncKeyState та подібні API мають конкретні сценарії застосування. Не опитуй систему без потреби.

Input loops
У іграх краще мати чіткий цикл вводу/оновлення/малювання.

Практика
72. Поясни, чому Timer control простіший за API SetTimer у звичайній формі.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

46. Системний трей та утиліти
Інтеграція із системним треєм зазвичай потребує Shell_NotifyIcon, структури NOTIFYICONDATA та обробки callback-повідомлення.

Архітектура
Створюєш невидимий або мінімізований host, додаєш icon, обробляєш клік, показуєш контекстне меню.

Дизайн
У трей-програмі завжди повинна бути зрозуміла команда виходу. Не залишай процес працювати безповоротно.

Практика
73. Спроєктуй трей-органайзер на папері перед кодом.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина VII. Графіка та ігри

47. Система координат
У PictureBox координати залежать від ScaleMode та налаштувань ScaleWidth/ScaleHeight.

Початок
Типово верхній лівий кут є початком системи координат.

Scale
Scale може змінювати діапазон, що дозволяє працювати не в пікселях, а у власних одиницях.

Практика
74. Намалюй осі X/Y і поділки.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

48. Line, Circle, PSet та PaintPicture
Ці методи формують просту 2D-графіку.

Приклади
Line малює відрізок, Circle — коло, PSet — точку, PaintPicture — копіює/малює зображення.

Колір
ForeColor, FillColor та DrawWidth змінюють вигляд.

Практика
75. Намалюй сітку 10×10 та простий графік функції.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

49. Анімація та game loop

Навчальна гра — це повторюваний цикл.

Стадії

1) input → 2) update → 3) collision → 4) render.

Delta time

У простому Timer-based проєкті можна рухати об’єкт на фіксовану кількість пікселів за tick. Для більш стабільної швидкості краще рахувати elapsed time та рухати залежно від часу.

Double buffering

Малюй у буфері/окремому PictureBox і показуй результат одним кроком, щоб зменшити мерехтіння.

Практичний приклад коду

```
Private BallX As Single
Private BallY As Single
Private VX As Single
Private VY As Single

Private Sub Form_Load()
    BallX = 100
    BallY = 100
    VX = 3
    VY = 2
    Timer1.Interval = 20
End Sub

Private Sub Timer1_Timer()
    BallX = BallX + VX
    BallY = BallY + VY

    If BallX < 0 Or BallX > Picture1.ScaleWidth Then VX = -VX
    If BallY < 0 Or BallY > Picture1.ScaleHeight Then VY = -VY

    Picture1.Cls
    Picture1.Circle (BallX, BallY), 8
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

76. Зроби кульку, яка відбивається від меж.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

50. Зіткнення, стан гри та простий Pong

Pong — чудова вправа на координати та стан.

Стан

BallX, BallY, VX, VY, PaddleY, ScoreLeft, ScoreRight.

Зіткнення

Перевіряй перетини прямокутників/радіусів, а потім змінюй швидкість.

Рендер

Очистити → намалювати → показати рахунок.

Практика

77. Додай керування двома ракетками.
78. Додай рахунок до 10 і перезапуск.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина VIII. Автоматизація та компоненти

51. Automation

Automation дозволяє керувати іншими COM-додатками через їхні об’єктні моделі.

Ідея

CreateObject("Excel.Application") створює automation-об’єкт, якщо Excel доступний.

Життєвий цикл

Відкрити → виконати → зберегти → Quit → встановити Nothing.

Помилки

Зовнішня програма може бути відсутня, зависнути або мати іншу версію. Обробляй ці випадки.

Практичний приклад коду

```
Dim xlApp As Object
Dim xlBook As Object

Set xlApp = CreateObject("Excel.Application")
Set xlBook = xlApp.Workbooks.Add
```

```
xlBook.Worksheets(1).Range("A1").Value = "Hello"
xlApp.Visible = True

' Після завершення контрольованого сценарію:
' xlBook.Close SaveChanges:=True
' xlApp.Quit
' Set xlBook = Nothing
' Set xlApp = Nothing
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

79. Напиши програму, яка відкриває Excel і записує A1.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

52. Word та Excel Automation

Excel зручний для таблиць, Word — для документів.

Excel

Workbooks, Worksheets, Range/Cells. Використовуй `Set` для об’єктних посилань.

Word

Documents, Selection/Range, Paragraphs.

Продуктивність

Великі обсяги краще передавати блоками/масивами, а не по одній клітинці в циклі.

Практика

80. Експортуй список клієнтів у Excel.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

53. COM

COM — компонентна модель, на якій побудовані багато старих Windows-технологій, включно з Automation.

Основи

Об’єкт має інтерфейс; клієнт взаємодіє через визначені методи/властивості.

ProgID/CLSID
Програмні ідентифікатори використовуються для створення/реєстрації COM-класів.

Сумісність
Зміна публічного інтерфейсу може ламати клієнтів. Для компонентів потрібно думати про versioning.

Практика
81. Поясни різницю між класом усередині EXE та COM-компонентом.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

54. ActiveX DLL
ActiveX DLL — окремий компонент із класами, які можуть використовувати інші програми.

Типовий сценарій
Business logic у DLL; UI в EXE.

Переваги
Розділення відповідальності, повторне використання, централізоване оновлення компонента за відповідних умов.

Складнощі
Реєстрація, binary compatibility, версії залежностей, deployment.

Практика
82. Спроєктуй Calculator DLL з методами Add/Subtract/Multiply/Divide.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

55. ActiveX EXE та OCX
ActiveX EXE — компонент у власному процесі; OCX — візуальний ActiveX control.

ActiveX EXE
Може бути out-of-process, що впливає на модель взаємодії та продуктивність.

OCX
Компонент UI із властивостями/методами/подіями, який можна вставляти в середовища, що підтримують ActiveX controls.

Коли не треба
Не створюй компонент, якщо маленький клас у звичайному проєкті вирішує задачу простіше.

Практика
83. Спроєктуй властивості та події власного графічного контролу.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

56. DLL і взаємодія компонентів

Є різниця між звичайною нативною DLL-інтеграцією та COM/ActiveX.

Декларації

VB6 може викликати функції DLL через Declare.

COM

COM працює через інтерфейси та реєстрацію.

Правило

Ніколи не копіюй випадковий Declare з форуму без перевірки сигнатури, розрядності та типів.

Практика

84. Склади таблицю: Declare vs COM vs Automation.

Що запам'ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина IX. Мережа та сучасні формати

57. Winsock

Winsock control дозволяє будувати старі TCP/UDP-застосунки в подієвій моделі.

TCP

Client/Server, Connect, DataArrival, SendData.

UDP

Без встановленого потоку з'єднання, але без гарантій TCP.

Практика

Для навчального чату зроби простий сервер і клієнт у локальній мережі. Не використовуй його безпечність як даність: протокол треба захищати на рівні застосунку.

Практичний приклад коду

```
Private Sub cmdConnect_Click()
    Winsock1.RemoteHost = "127.0.0.1"
    Winsock1.RemotePort = 5000
    Winsock1.Connect
End Sub
```

```
Private Sub Winsock1_Connect()
    Winsock1.SendData "HELLO|" & Text1.Text
End Sub

Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)
    Dim S As String
    Winsock1.GetData S, vbString
    List1.AddItem S
End Sub
```

Примітка: Після запуску зміни хоча б одну річ у прикладі: ім’я контролу, умову, число, формат або текст повідомлення. Мета — зробити код своїм, а не лише скопіювати його.

Практика

85. Опиши протокол для текстового чату: команда, довжина, payload.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

58. HTTP на концептуальному рівні

HTTP — протокол запит-відповідь. VB6 не має сучасного вбудованого HTTP-клієнта рівня нових мов, але через COM-бібліотеки можна працювати з XMLHTTP/WinHTTP за наявності відповідних компонентів.

Запит

Method, URL, headers, body.

Відповідь

Status code, headers, body.

Надійність

Таймаути, перевірка статусу, обмеження розміру відповіді та коректне кодування.

Практика

86. Побудуй псевдокод GET-запиту та розбору відповіді.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

59. XML та JSON

Дані часто приходять у структурованому форматі.

XML
Дерево вузлів, атрибути, текст. Зручний для старих COM XML-парсерів.

JSON
Об’єкти, масиви, рядки, числа, Boolean, null. У VB6 зазвичай потрібен сторонній або самописний parser.

Правило
Не парси JSON простими InStr, якщо формат може бути складним: вкладені об’єкти, екранування, Unicode.

Практика
87. Опиши структуру JSON для клієнта.
88. Склади просту XML-модель замовлення.

Що запам’ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

60. Кодування, Unicode та сумісність
Для текстових програм критично розуміти різницю між ANSI-кодуваннями та Unicode.

VB6 String
Внутрішньо VB6 використовує Unicode/BSTR для String, але старі API та файлові операції можуть працювати через ANSI-інтерфейси.

Файли
Не припускай, що `Open For Input` автоматично вирішить кодування UTF-8 як сучасні бібліотеки. Для UTF-8 часто потрібен явний механізм конвертації/ADODB.Stream або API.

Символи
Перевіряй, як дані проходять через UI → файл → API → БД.

Практика
89. Зроби тест із українським текстом і перевір його збереження/відкриття.

Що запам’ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Частина X. Налагодження і великі проєкти

61. Debugger
Налагодження — не «пошук магії», а системний процес.

Інструменти
Breakpoint, Step Into, Step Over, Step Out, Locals, Watch, Immediate Window, Call Stack.

Debug.Print
Виводь ключові значення в Immediate Window.

Стратегія
Відтворити → локалізувати → мінімізувати → виправити → перевірити регресію.

Практика
90. Знайди навмисну логічну помилку в циклі.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

62. Оптимізація
Спочатку правильність, потім швидкість.

Типові вузькі місця
Зайві цикли, частий доступ до диска, поелементна Automation Excel, надмірне перерисовування UI, непараметризовані важкі SQL-запити.

Вимірювання
Вимірюй до і після оптимізації. Не оптимізуй код лише тому, що він «виглядає повільним».

Пам'ять
Великі масиви та об'єкти потребують контролю життєвого циклу.

Практика
91. Знайди 3 місця, де можна кешувати дані.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

63. Архітектура
Для великих VB6-проєктів потрібні правила структури.

Розділення
UI / Business Logic / Data Access / Infrastructure.

Імена
frmMain, modDatabase, clsCustomer — угода важливіша за конкретний стиль.

Залежності
Форма не повинна знати деталі SQL-тексту, якщо це можна винести в data layer.

Практика
92. Намалюй архітектуру CRM з 4 шарами.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

64. Безпечна робота з вводом
Старі desktop-програми теж повинні захищати дані.

Ввід
Trim, Length, допустимі символи, діапазони.

SQL
Параметри замість конкатенації.

Файли
Ніколи не видаляй довільний шлях тільки тому, що його передали в TextBox.

Секрети
Не записуй паролі та токени у відкритий лог.

Практика
93. Склади чекліст безпечного вводу для менеджера клієнтів.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

65. Пакування та розповсюдження
Компіляція в EXE — тільки частина доставки програми.

Залежності
VB6 runtime, ActiveX/OCX/DLL, COM-реєстрація, database provider.

Версії
Зафіксуй версії компонентів і перевір інсталяцію на чистій тестовій машині.

Логи
Для реального застосунку передбачай журналювання та шлях до логів.

Практика
94. Склади deployment checklist із 15 пунктів.

Що запам'ятати
Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

66. Підсумковий проєкт: менеджер клієнтів

Фінальний проєкт об’єднає UI, класи, файли, SQL/ADO, обробку помилок та експорт.

Функції

Список клієнтів, пошук, додавання, редагування, видалення, контактні дані, нотатки, експорт.

Модель

clsCustomer + Repository/Data Access + frmMain + frmCustomer.

Тестування

Порожній ввід, дубльовані записи, відсутня БД, помилка підключення, неправильна дата, закриття форми під час операції.

Розвиток

Додати ролі, журнал змін, резервне копіювання, імпорт CSV, Excel-експорт, фільтри.

Практика

95. Зроби мінімальну версію.
96. Виділи Data Access в окремий модуль/клас.
97. Додай логування помилок.
98. Зроби резервну копію БД перед масовим видаленням.

Що запам’ятати

Головна ідея розділу: спочатку зрозумій модель і дані, потім пиши найменший код, який це реалізує. Після цього тестуй окремі випадки, включно з неправильним вводом та порожніми даними.

Довідник: основні типи даних VB6

Тип	Призначення	Приклад
Boolean	Логічне значення	True
Byte	0..255	255
Integer	Ціле 16-бітове	1000
Long	Ціле 32-бітове	100000
Single	Дробове, менша точність	1.25
Double	Дробове, вища точність	3.14159
Currency	Грошові значення	1250.50
Date	Дата/час	Date
String	Текст	"Hello"
Object	Об’єктне посилання	Set X = Form1
Variant	Універсальне значення	Dim V As Variant

Довідник: часто вживані елементи VB6

Контрол	Для чого	Ключові властивості/методи
Form	Вікно/контейнер	Caption, Visible, Enabled, Show, Hide
Label	Текст	Caption
TextBox	Ввід/редагування тексту	Text, MaxLength, MultiLine, SetFocus
CommandButton	Команда	Caption, Enabled, Click
CheckBox	Так/ні	Value
OptionButton	Один варіант у групі	Value
Frame	Групування	Caption
ListBox	Список	AddItem, RemoveItem, ListIndex
ComboBox	Комбінований список	AddItem, Text, ListIndex
PictureBox	Графіка/контейнер	Cls, Line, Circle, PSet, PaintPicture
Image	Зображення	Picture, Stretch
Timer	Періодичні події	Interval, Timer
CommonDialog	Системні діалоги	ShowOpen, ShowSave, ShowColor, ShowFont

Довідник: контрольний список перед релізом

99. У всіх модулях увімкнено Option Explicit.
100. Немає невикористаних глобальних змінних без причини.
101. Усі файли закриваються після роботи.
102. З’єднання з базою та Recordset закриваються.
103. Є обробка очікуваних помилок.
104. Введення перевіряється.
105. Небезпечні SQL-конкатенації прибрані або локалізовані.
106. Є резервна копія даних там, де видалення незворотне.
107. Логи не містять паролів та секретів.
108. Програма перевірена на чистому середовищі з усіма залежностями.
109. Компоненти та OCX/DLL мають зафіксовані версії.
110. У користувача є зрозумілий спосіб завершити програму.

Фінальний навчальний маршрут

Коли базовий синтаксис уже знайомий, не намагайся вивчити все одночасно. Рухайся хвилями:

Етап 1 — мова
змінні → умови → цикли → масиви → рядки → процедури → помилки

Етап 2 — GUI
Form → TextBox → ListBox/ComboBox → меню → діалоги → Timer → графіка

Етап 3 — дані
файли → класи → ADO/SQL → CRUD → транзакції

Етап 4 — Windows
API → повідомлення → Clipboard → Shell → Registry → трей

Етап 5 — професійний рівень
Automation → COM → ActiveX DLL/EXE → розподіл на шари → deployment

Висновок
VB6 вчить корисної дисципліни: чітко називати дані, розуміти життєвий цикл об’єкта, розділяти UI та логіку, перевіряти помилки та контролювати зовнішні залежності. Навіть якщо згодом перейти на іншу мову, ці принципи залишаються актуальними.

Найкраща перевірка знань — не тест, а власна програма. Після цієї книги зроби хоча б один завершений проєкт: блокнот, менеджер файлів, каталог, облік клієнтів або просту гру. Завершені маленькі проєкти дають більше навичок, ніж десятки прочитаних прикладів.